



23 października 2003

Wykorzystanie XML-a we własnych aplikacjach

Modele dostępu do dokumentu XML

Pozwalają programistom na dostęp do zawartości dokumentów XML na wysokim poziomie:

- korzystamy z abstrakcyjnych obiektów,
- nie troszczymy się o analizę leksykalną i składniową.

SAX – Simple API for XML:



- model zdarzeniowy.

DOM – Document Object Model:

- obiektywny model drzewa struktury,
- zwykle implementowany przy użyciu SAX.

XML data binding – wiązanie XML-a.

Pull parsing – przetwarzanie strumieniowe.

Implementacja modelu – parser

Parser – moduł programistyczny analizujący dokument XML i udostępniający jego zawartość w postaci abstrakcyjnego modelu.


Funkcjonalność parsera:

- analiza leksykalna i składniowa,
- sprawdzenie poprawności strukturalnej (tylko parser walidujący).

Generyczność – niezależność od konkretnego języka!

Po co abstrakcyjne modele:

- jednolity sposób programowania, niezależnie od użytego parsera,
- możliwość wymiany parsera,
- możliwość porównywania parserów.




SAX – Simple API for XML

Idea:

- dokument XML jako ciąg zdarzeń,
- program reaguje na wybrane zdarzenia.

Status:

- 1998: SAX 1.0,
- 2000: SAX 2.0 – najważniejsze rozszerzenia:
 - obsługa przestrzeni nazw,
 - cechy (features) - wartości boolowskie,
 - właściwości (properties) - dowolne obiekty,
 - zmiany nazw wielu obiektów.




Działanie modelu SAX – przykład

```
<?xml version="1.0"?>
<wiersz bialy="nie">
  <autor>
    William Shakespeare
  </autor>
</tytul>
</wiersz>
```

Parser ← setDocumentHandler

Aplikacja

```
startDocument()
startElement("wiersz", [bialy="nie"])
ignoreableWhitespace(spacje)
startElement("autor", [])
characters("William...")
endElement("autor")
ignoreableWhitespace(spacje)
throw SAXException
```



SAX2 – pakiet org.xml.sax

Interfejsy implementowane przez parser:

XMLReader

- parse (2 metody),
- setContentHandler,
- ...

Attributes

- getLength,
- getLocalName, getQName,
- getValue (2 metody).


Opcjonalny: Locator

Interfejsy implementowane przez użytkownika parsera:

ContentHandler – zdarzenia:

- characters,
- ignoreableWhitespace,
- startDocument, endDocument,
- startElement, endElement,
- processingInstruction,
- setDocumentLocator.

ErrorHandler, DTDHandler, EntityResolver.



SAX2 – pakiet org.xml.sax

Standardowa klasa:
org.xml.sax.InputSource –
może pobierać dane z
InputStream, Reader, String.

Wyjątek:
SAXException – podnoszony w
przypadku wystąpienia błędu.

Klasy pomocnicze (pakiet
org.xml.sax.helpers):

- DefaultHandler –
implementujemy podklasy tej
klasy,
- XMLReaderFactory,
- AttributesImpl,
- LocatorImpl.



SAX – kroki implementacji

Tworzymy podklasę klasy org.xml.sax.helpers.DefaultHandler.
Pobieramy obiekt org.xml.sax.XMLReader z fabryki.
Rejestrujemy stworzoną podklasę w parserze (XMLReader) metodami
set...Handler.
Wywołujemy metodę parse.



Filtry SAX

Implementują interfejs org.xml.sax.XMLFilter.
Rozszerzają klasę org.xml.sax.helpers.XMLFilterImpl.
Specyficzne implementacje interfejsów: ContentHandler, DTDHandler,
EntityResolver, ErrorHandler.

Można je łączyć w łańcuchy:

```
XMLReader reader;  
...  
XMLFilterImpl f1 = new XMLFilterImpl(reader);  
XMLFilterImpl f2 = new XMLFilterImpl(f1);  
f2.parse(...);
```



Model DOM

Dostęp do całego dokumentu (HTML lub XML), z wyjątkiem DTD.
Części składowe:

- DOM Level 1 (październik 1998):
 - podstawowe metody dostępu do struktury dokumentu,
- DOM Level 2 (listopad 2000):
 - nowe cechy XML-a, np. przestrzenie nazw,
 - Views – "widoki" dokumentu po zastosowaniu stylów CSS,
 - Events – obsługa zdarzeń,
 - Style – manipulowanie arkuszami stylów,
 - Traversal and Range – "podróżowanie" po dokumencie XML.
- DOM Level 3 (w przygotowaniu):
 - Load and Save – ładowanie i zapisywanie dokumentu,
 - Validation – dostęp do definicji struktury dokumentu (DTD),
 - XPath – dostęp do węzłów DOM przez wyrażenia XPath.



DOM Core

Bazowa część specyfikacji DOM.

Umożliwia:

- budowanie dokumentów,
- nawigację po strukturze dokumentów,
- dodawanie elementów i atrybutów,
- modyfikacje elementów i atrybutów,
- usuwanie elementów/atributów i ich zawartości.

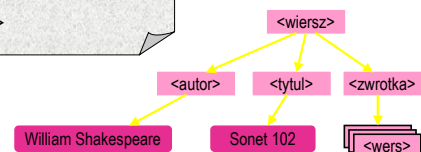
Wady:

- pamięciożerność,
- niska efektywność.

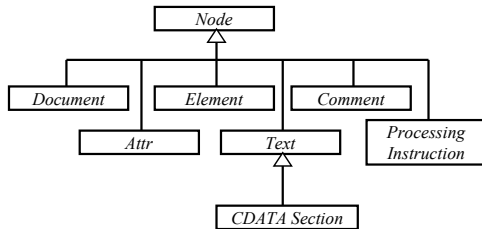


Drzewo DOM - przykład

```
<?xml version="1.0"?>  
<wiersz>  
  <autor>William Shakespeare  
</autor>  
  <tytul>Sonet 102</tytul>  
  <zwrotka  
    <wers>...</wers>  
    ...  
  </zwrotka>  
</wiersz>
```



DOM – najważniejsze interfejsy



Interfejs Node

Dostęp do zawartości:

- `getAttributes()`
- `getChildNodes()`
- `getFirstChild()`
- `getLastChild()`
- `getNextSibling()`
- `getPreviousSibling()`
- `getNodeName()`
- `getNodeValue()`
- `getNodeType()`
- `getOwnerDocument()`
- `getParentNode()`
- `hasChildNodes()`

Manipulacja zawartością:

- `appendChild(Node)`
- `insertBefore(Node, Node)`
- `removeChild(Node)`
- `replaceChild(Node, Node)`
- `setNodeValue(String)`
- `setNodeName(String)`

Klonowanie:

- `cloneNode(boolean)`

Klasy pomocnicze DOM

NamedNodeMap:

- tablica haszująca obiektów Node (np. atrybutów).

NodeList:

- wektor obiektów Node (np. dzieci danego węzła).

DOMException:

- wyjątek podnoszony w przypadku błędnej modyfikacji węzła.

SAX ↔ DOM

- Przetwarzanie wsadowe.
- Oszczędny czasowo i pamięciowo.
- Dobry do wyławiania z dokumentu wybranych elementów.
- Dokument tylko do odczytu
- Całe drzewo dokumentu ładowane do pamięci.
- Kosztowny czasowo i pamięciowo.
- Pozwala wędrować po drzewie dokumentu.
- Pozwala tworzyć i modyfikować dokumenty.

Parsery XML – przegląd

Java:

- XP Jamesa Clarka (niewalidujący).
- Xerces (dostępny także dla C++ i Perla, XML Schema, SAX 1.0/2.0, DOM level 1 i 2).
- XML4J – IBM XML Parser for Java (XML Schema, SAX 1.0/2.0, DOM level 1, 2, 3).
- Oracle XML Parser for Java (walidujący).
- ... kilkadziesiąt innych.

C, C++:

- XML4C – IBM XML Parser for C++ (XML Schema, SAX 1.0/2.0, DOM level 1, 2, 3, eksperymentalna implementacja XML 1.1).
- Expat Jamesa Clarka (niewalidujący).

Parsery XML – przegląd

Perl:

- `XML::Parser` – pakiet wykorzystujący parser Expat J. Clarka napisany w C.
- `XML::DOM`.

Python:

- PyXML.

Microsoft XML Core Services (MSXML 4.0, komponent COM):

- możliwość dostępu z różnych języków programowania: ECMAScript, Java, Perl, Python, SQL, VisualBasic.

XML Data Binding

Dokumenty XML a obiekty Javy:

- DTD/schemat odpowiada definicji klasy,
- dokument XML (instancja schematu) odpowiada obiektowi (instancji klasy).

Pomysly:

- automatyczne generowanie klas z DTD/schematów,
- generowane klasy implementują serializację (i nic więcej).

Implementacje:

- Dynamic XML, ObjectSpace,
- JAXB – Java Architecture for XML Binding, Sun Microsystems,
- Castor, Exolab,
- ...



Przykład: DXML

```
<!ELEMENT Phonebook (Person*)>
<!ELEMENT Person
  (Name, Phone*)>
<!ELEMENT Name
  (Firstname, Lastname)>
<!ELEMENT Firstname (#PCDATA)>
<!ELEMENT Lastname (#PCDATA)>
<!ELEMENT Phone (#PCDATA)>

public interface IPerson extends
com.objectspace.xml.IDXMLInterface {
// element Name
public IName getName();
public void setName(IName arg0);
// element Phone
public void addPhone(String arg0);
public int getPhoneCount();
public void setPhones(Vector arg0);
public String[] getPhones();
public void setPhones(String[] arg0);
public Enumeration getPhoneElements();
public String getPhoneAt(int arg0);
public void insertPhoneAt
  (String arg0, int arg1);
public void setPhoneAt
  (String arg0, int arg1);
public boolean removePhone
  (String arg0);
public void removePhoneAt(int arg0);
public void removeAllPhones();
}
```



DXML: jak z tego korzystać

Przygotowanie DTD.

Wygenerowanie klas.

Korzystanie w kodzie w Javie:

```
import com.objectspace.xml.*;
...
xmlDocument = Xml.openDocument(new File("phonebook.xml"));
IPhonebook phonebook = (IPhonebook) xmlDocument.getRoot();
```



Przetwarzanie strumieniowe – pull parsing

Alternatywa dla modelu SAX:

- aplikacja "wyciąga" kolejne zdarzenia z parsera,
- przetwarzanie kontrolowane przez aplikację, a nie parser,
- parser działa podobnie jak iterator, kursor lub strumień danych,
- zachowane cechy modelu SAX:
 - duża wydajność,
 - możliwość przetwarzania dowolnie dużych dokumentów.

Dostępne implementacje:

- .Net XmlReader, Microsoft,
- XmlPull API i jego implementacje (XNI 2 XmlPull, XPP3/MXP1),
- BEA XML Stream API.

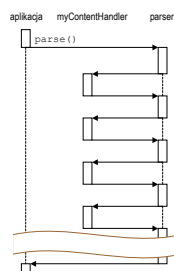
Standaryzacja:

- Java Community Process, JSR 173: Streaming API for XML.

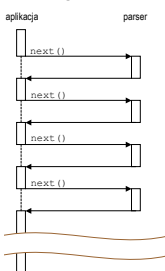


SAX a Pull parsing

SAX:



Pull parsing:



Pull parsing – korzyści

Jeszcze większa wydajność niż w (i tak już wydajnym) modelu SAX, dzięki:

- możliwości przerywania przetwarzania przed końcem pliku, gdy potrzebujemy z niego tylko część danych,
- możliwości zmniejszenia liczby kopii obiektów typu String,
- szybszemu filtrowaniu zdarzeń.

Możliwość prostej obróbki wielu dokumentów jednocześnie.

Bardziej „proceduralny” styl programowania, co daje:

- mniej stanów do pamiętania,
- możliwość użycia rekursji,
- zwiększone powtórne użycie kodu (reusability).

Źródło: M. Piechawski, "Nie pozwól się popychać", Software 2.0, 6/2003



SAX czy pull parsing – co wybrać?

Pull parsing sprawdza się, gdy:

- kończymy przetwarzanie po wystąpieniu poszukiwanych danych,
- przetwarzanie zdarzenia zależy od kontekstu (np. od tego, czy jesteśmy wewnątrz pewnego elementu),
- przetwarzamy równoległe więcej niż jeden plik (np. porównujemy pliki).

SAX sprawdza się, gdy:

- chcemy odfiltrować dokument – interesują nas tylko wybrane elementy
- za jednym przebiegiem dokonujemy kilka niezależnych rodzajów przetwarzania.



Gdzie szukać dalej

SAX Home Page:

www.saxproject.org

Document Object Model (DOM):

www.w3.org/DOM

xml.coverpages.org/dom.html

Common API for XML Pull Parsing

www.xmlpull.org

Paweł Gajda, SAX i DOM, czyli XML w naszych aplikacjach

www.empolis.pl → Osiągnięcia → Archiwum publikacji

□ Software 2.0, nr 6/2001, Wydawnictwo Software

Tomasz Brauncajs, JAXB i Castor – wiązanie XML-a w Javie

□ Software 2.0, nr 6/2002, Wydawnictwo Software

Michał Plechawski, Nie pozwól się popychać

□ Software 2.0, nr 6/2003, Wydawnictwo Software



Gdzie szukać dalej

IBM alphaWorks:

www.alphaworks.ibm.com

Free XML tools and software – Lars Marius Garshol:

www.garshol.priv.no/download/xmltools/

XML w Javie:

java.sun.com/xml

